Integrating ProB into the TLA Toolbox

Dominik Hansen, Jens Bendisposto, Michael Leuschel

Institut für Informatik, Universität Düsseldorf**
{hansen, bendisposto, leuschel}@cs.uni-duesseldorf.de

The TLA2B translator [2] enables the validation of TLA$^+$ specifications with the model checker, animator and constraint-based checker ProB [4]. In order to provide a convenient way to use ProB as a new validation tool for TLA$^+$, we integrated ProB into the TLA toolbox (Fig. 1).

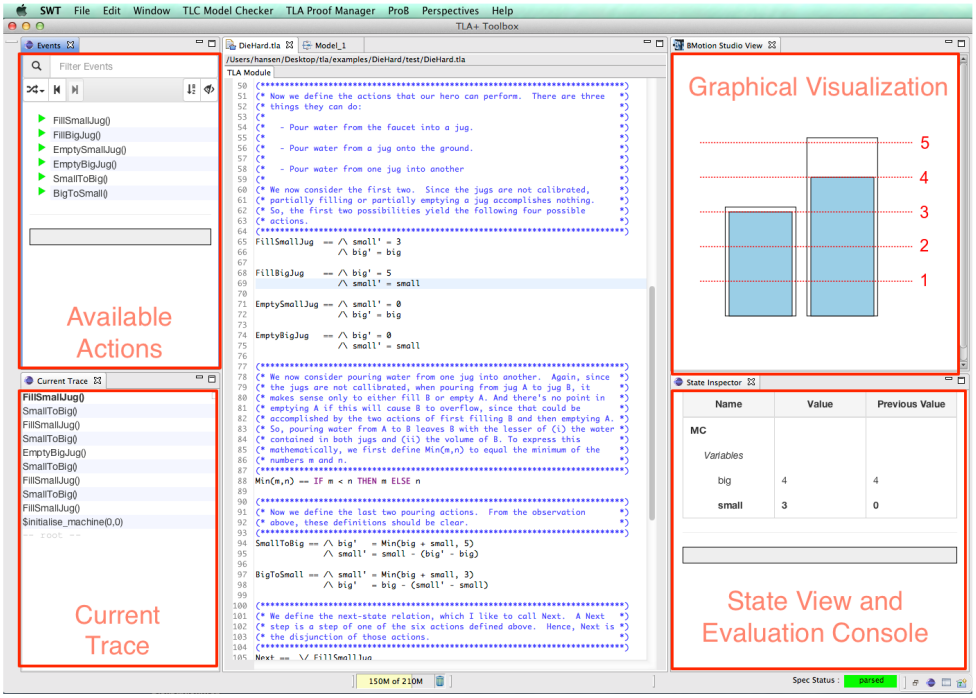

Fig. 1. Using ProB within the TLA Toolbox

Features. Animation is one of the most useful features of ProB that is not currently supported by the TLA$^+$tool chain. It allows a user to interact with the specification by discovering the statespace step by step and to evaluate expressions on the current state. Animation could be especially useful for newcomers to get familiar with TLA$^+$. Moreover, the ProB animator can be used to control a graphical visualization of a specification (top right view in Fig. 1). Apart from that, ProB is a model checker complementary to TLC. Experimental results comparing both model checkers can be found in [3]. Another useful feature of ProB is called constrained based model checking. In this mode of operation, ProB does not explore all reachable states starting from the initial state(s), but checks whether a single step of the next-state relation can result in an invariant violation

** Part of this research has been sponsored by the EU funded FP7 project 287563 (ADVANCE).

independently of the initial state(s) of the model. If the constraint based checker finds a counter-example, this indicates that the model may contain a problem. The step discovered by the constraint based checker leads from a valid state (satisfying the invariant(s)) to an invariant violation. Note that the valid state is not necessarily reachable from a valid initial state, but this situation indicates that the invariant(s) are not inductive and cannot be proved.

Integration into the toolbox. We already had an integration for the Eclipse based RODIN [1] platform. Furthermore, our architecture was already decomposed, i.e., the small part that depends on RODIN only contained declarative bindings for the UI (e.g., where menus and popups are shown) and the code for loading an EventB model. The rest of the tool including the implementation of the UI was already independent from RODIN and could be reused. We developed a plug-in that contains the UI bindings for the toolbox and the code for loading TLA$^+$ models. We also needed to make small changes to the product and target definitions of the toolbox (i.e, to include ProB in the toolbox).

TLA2B. We developed a new version of the TLA2B translator especially for the toolbox integration. The translator is build upon the SANY parser and uses a TLC run configuration provided by the toolbox. In contrast to the old translator, the new one directly creates the abstract syntax tree (AST) ProB is working on. Hence, we skip the parsing procedure of the formerly created B machine. Directly creating the AST makes us more flexible because we do not have to extend the B languages for the missing TLA$^+$ constructs (e.g. the CHOOSE operator). Moreover, we need no longer a renaming phase handling naming clashes. The AST also contains nodes for constructs from other formal languages such as Z or Event-B which are also supported by ProB.

Limitations and current work. A main difference between TLA$^+$ and B are the concepts of typing. While TLA$^+$ is untyped, B is strongly typed. Since the ProB kernel was designed and optimized to support a typed language, TLA2B contains a type inference algorithm for TLA$^+$. All data values of TLA$^+$ are supported including TLC's model values. However, there are some resulting restrictions such as values of different types can not be mixed in a set and variables can not have a polymorphic type. While these limitations seems be unavoidable, we are working on the other issues such as the support of recursive definitions. Even a support of temporal formulas seems be possible since ProB has a integrated LTL model checker. Moreover, we are currently working on a closer interaction of ProB and TLC: ProB could be used to setup the constants for TLC (i.e, creating a run configuration) or automatically replaying-counter examples produced by TLC in the animator.

At the workshop we want to present the current state of the integration of ProB into the toolbox, in the hope of getting feedback and to guide further development.

References

1. J.-R. Abrial, M. Butler, and S. Hallerstede. An open extensible tool environment for Event-B. In Z. Liu and J. He, editors, *Proceedings ICFEM'06*, LNCS 4260, pages 588–605. Springer-Verlag, 2006.
2. D. Hansen and M. Leuschel. Translating TLA+ to B for validation with ProB. In *Proceedings iFM'2012*, LNCS 7321, pages 24–38. Springer, June 2012.
3. D. Hansen and M. Leuschel. Translating B to TLA+ for validation with TLC. Technical report, Institut für Informatik, Universität Düsseldorf, 2013. To appear in ABZ'14.
4. M. Leuschel and M. J. Butler. ProB: an automated analysis toolset for the B method. *STTT*, 10(2):185–203, 2008.